Shortest-Path Based Column Generation on Extremely Large Networks with Many Resource Constraints

Faramroze G. Engineer,
George L. Nemhauser and Martin W.P. Savelsbergh

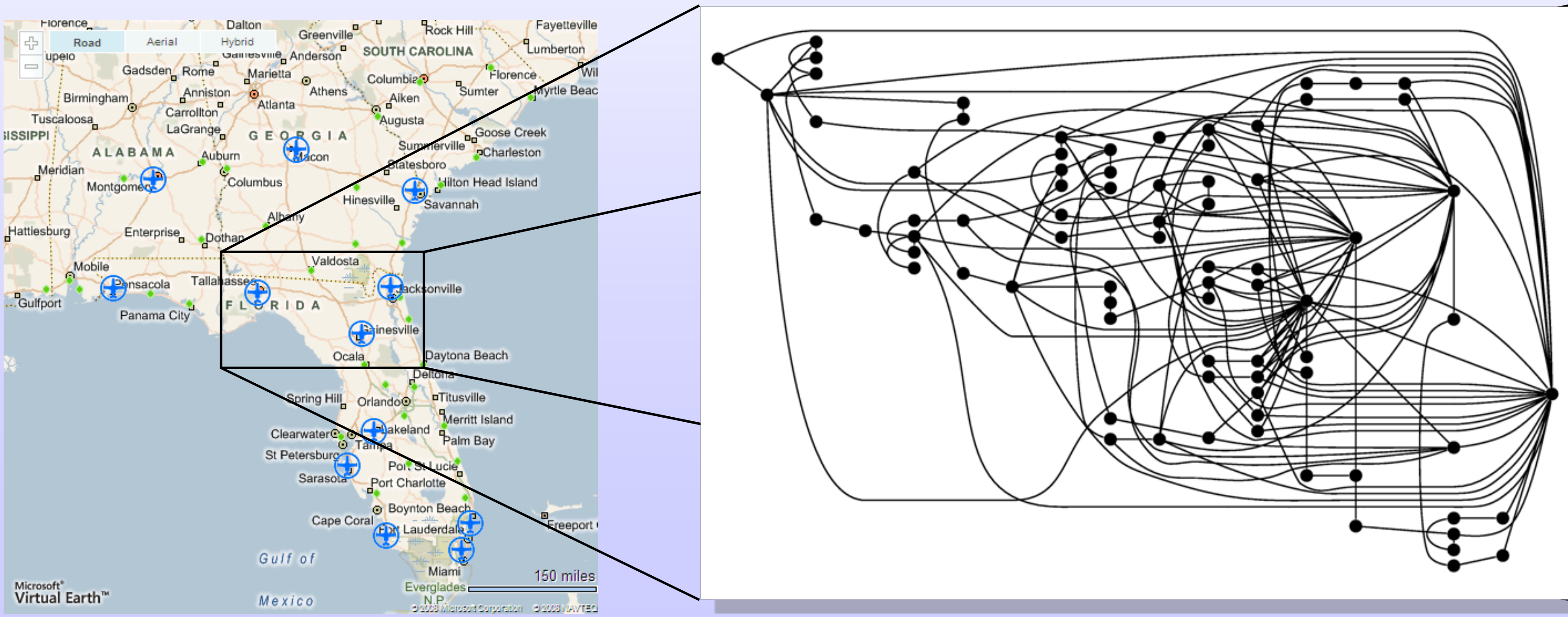
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

Column Generation, 2008

Resource Constrained Shortest Paths (RCSPP)

Georgia Institute of Technology
The H. Milton Stewart School of Industrial and Systems Engineering



Outline

- RCSPP preliminaries:
  - problem description and
  - standard DP techniques.
- Strengthening of dominance criteria using support structures.
- An arc-based relaxation of RCSPP.
- An iterative relaxation + bounding scheme.
- Preprocessing by aggregation.
- Computational results on networks > 10^6 nodes, arcs and resource constraints.

Problem Description

Given network $\mathcal{N} = (N,A)$, where :

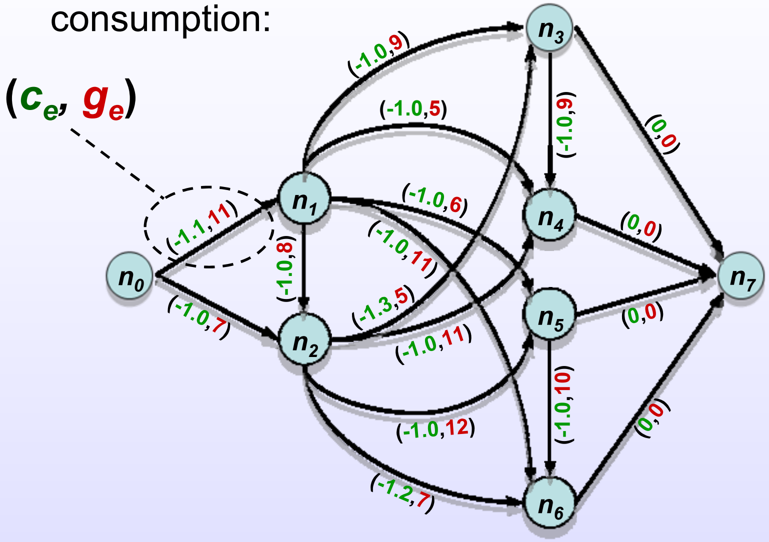
- n^s and n^t are the source and sink nodes respectively,
- c_e – cost of traversing arc e,
- g_e^k – non-negative integer amount of resource $k \in \{1,\ldots,K\}$ consumed when traversing arc e, and
- b^k – amount of resource k that is available,

find P^* s.t.

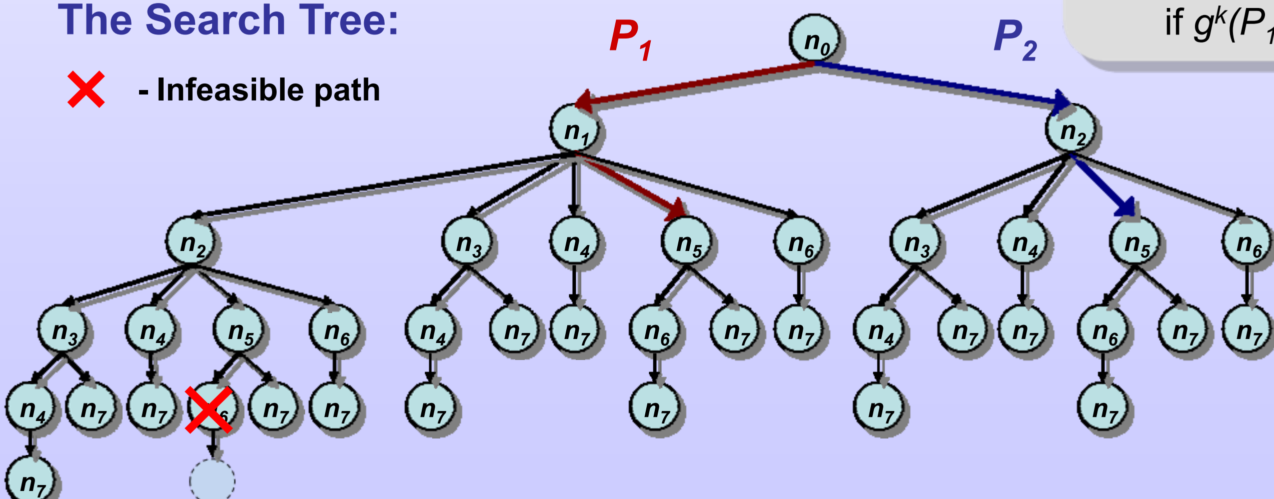
$$P^* \in \operatorname{argmin}\left\{ c(P) = \sum_{e \in A(P)} c_e : \begin{array}{l} P \text{ is a path from } n^s \text{ to } n^t \text{, and} \\ g^k(P) = \sum_{e \in A(P)} g_e^k \leq b^k, \forall k \in \{1,\ldots,K\} \end{array} \right\}$$

Solving RCSPP using DP

Example: Find shortest path from n_0 to n_7 that does not exceed 40 units of resource consumption:



The Search Tree:



The DP algorithm

Enumerate possible paths in a tree search while checking for:

1. **FEASIBILITY**, and
2. **DOMINANCE**.

Dominance

Given feasible paths P_1 and P_2 from source to node n, P_1 dominates P_2 if:

1. $c(P_1) \leq c(P_2)$, and
2. any feasible extension of P_2 to the sink is also feasible for P_1. Typically recognized if $g^k(P_1) \leq g^k(P_2)$ for all $k = 1,\ldots,K$.

$P_1 = (n_0, n_1, n_5)$

$c(P_1) = -2.1$ and $g(P_1) = 17$

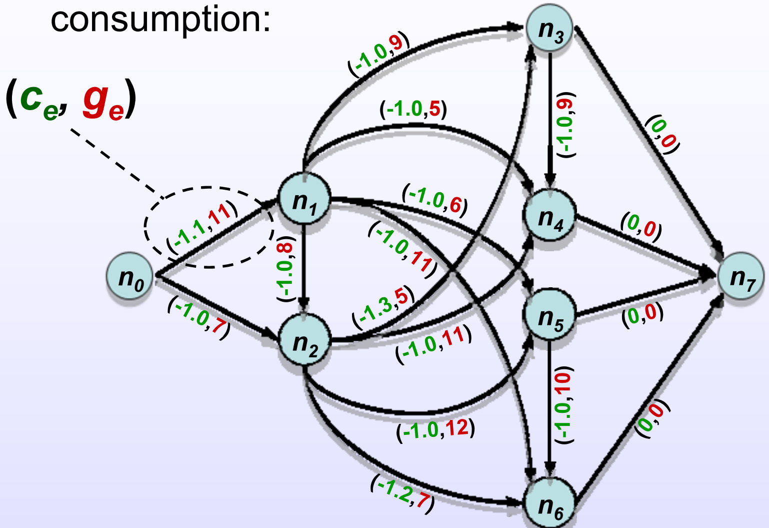
$P_2 = (n_0, n_2, n_5)$

$c(P_2) = -2.0$ and $g(P_2) = 19$

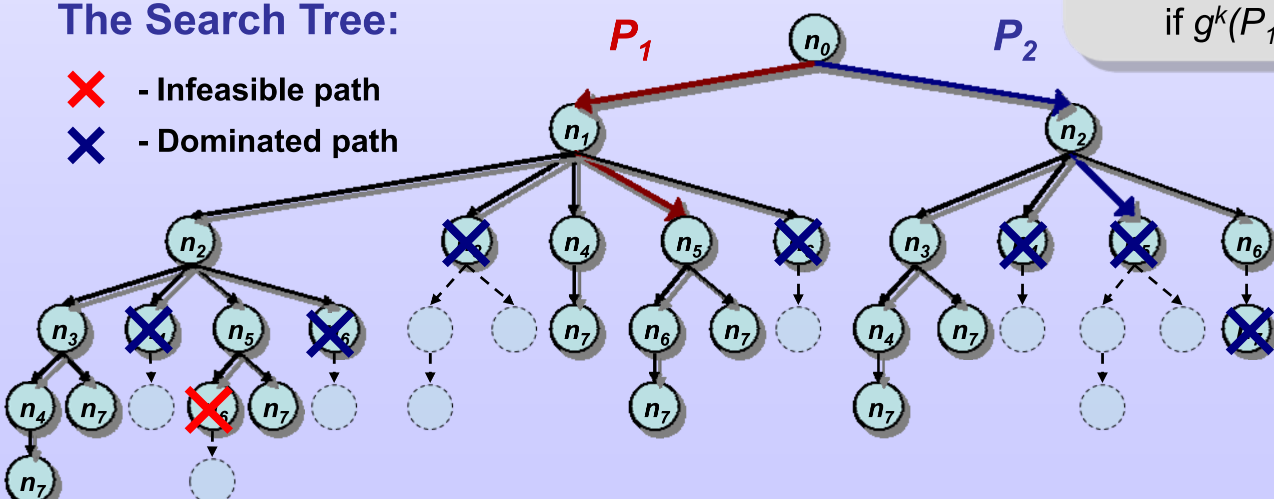
$\Rightarrow$ P_1 dominates P_2

Solving RCSPP using DP

Example: Find shortest path from n_0 to n_7 that does not exceed 40 units of resource consumption:



The Search Tree:



$P_1 = (n_0, n_1, n_5)$

$c(P_1) = -2.1$ and $g(P_1) = 17$

$P_2 = (n_0, n_2, n_5)$

$c(P_2) = -2.0$ and $g(P_2) = 19$

$\Rightarrow$ **P_1 dominates P_2**

The DP algorithm

Enumerate possible paths in a tree search while checking for:

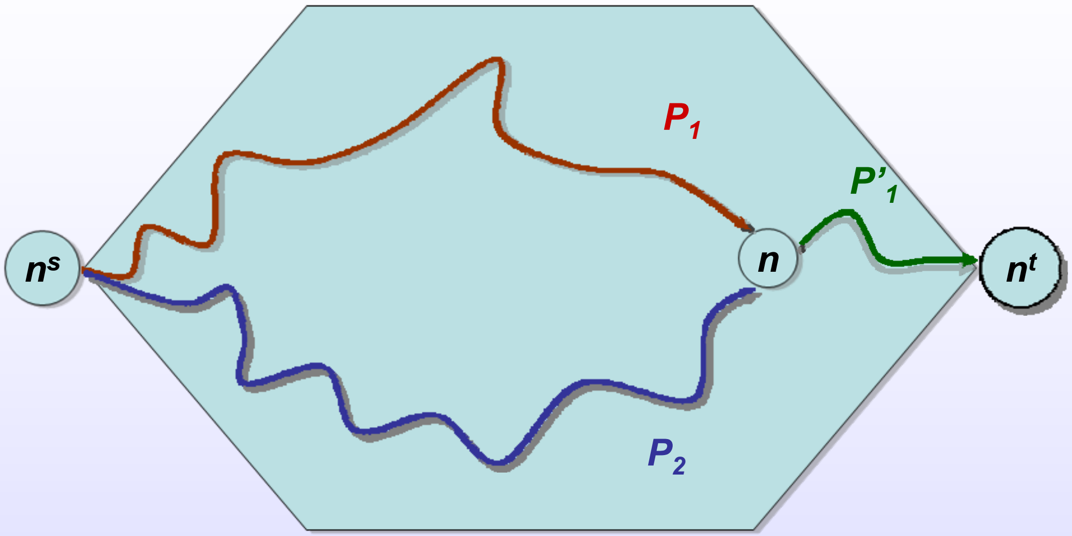
1. **FEASIBILITY**, and
2. **DOMINANCE**.

Dominance

Given feasible paths P_1 and P_2 from source to node n, P_1 dominates P_2 if:

1. $c(P_1) \leq c(P_2)$, and
2. any feasible extension of P_2 to the sink is also feasible for P_1. Typically recognized if $g^k(P_1) \leq g^k(P_2)$ for all $k = 1,\ldots,K$.

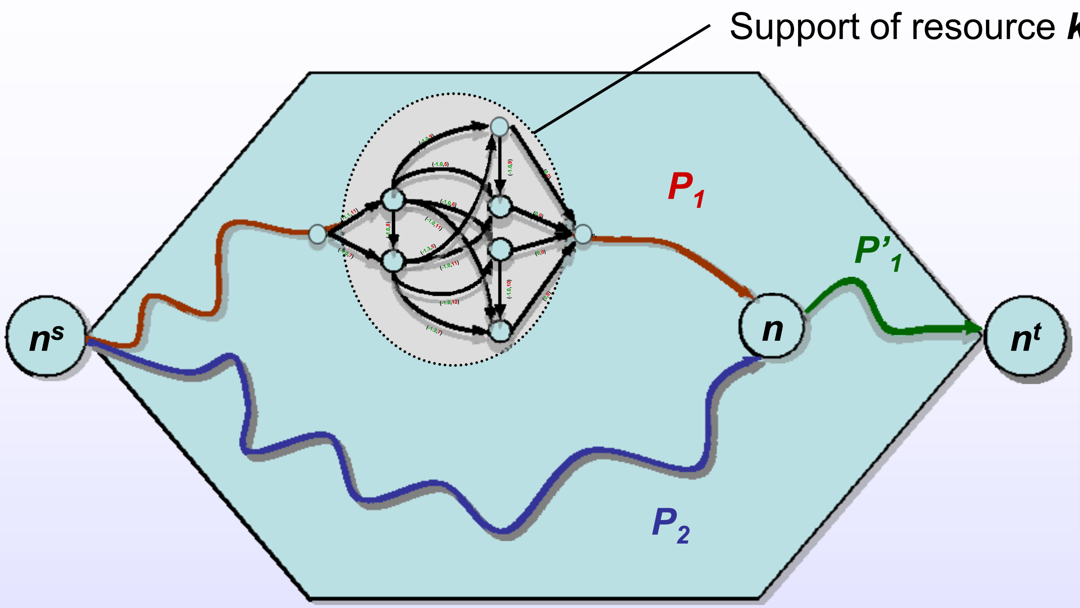
Strengthening Dominance Criteria



If $c(P_1) < c(P_2)$ but $g^k(P_1) > g^k(P_2)$, then we cannot say P_1 dominates P_2.

However, if $g^k(P_1) + g^k(P'_1) \leq b^k$ for all P'_1 from n to the sink, then we do not need to check dominance with respect to resource k.

Strengthening Dominance Criteria



Support of resource k

$N^k \subseteq N$ is a support of resource k if it contains all nodes $n \in N$ for which there exists path P_1 from source to node n and P'_1 from n to sink with:

1. $g^k(P_1) + g^k(P'_1) > b^k$
2. $g^k(P_1) > 0$, and
3. $g^k(P'_1) > 0$.

If $c(P_1) < c(P_2)$ but $g^k(P_1) > g^k(P_2)$, then we cannot say P_1 dominates P_2.

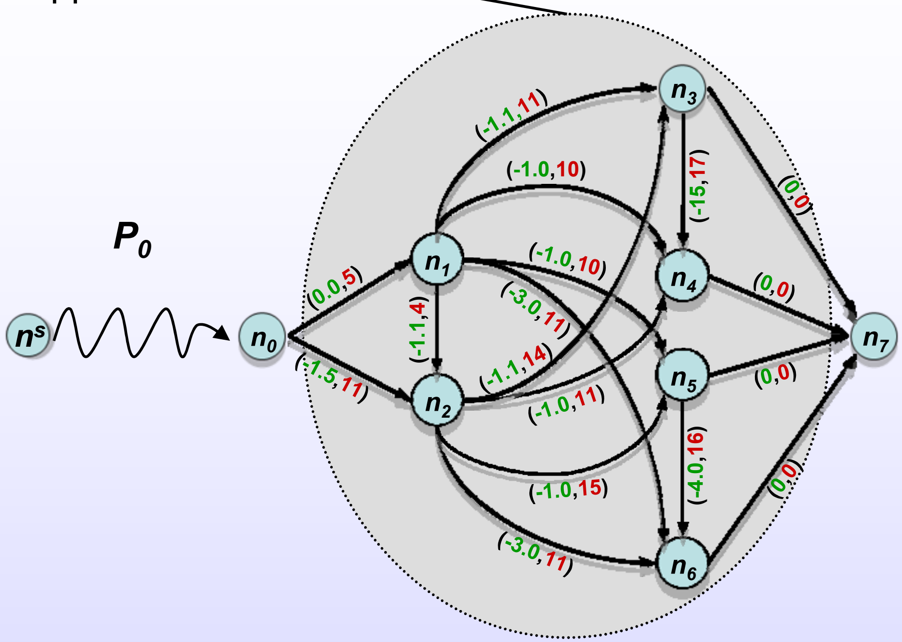
However, if $g^k(P_1) + g^k(P'_1) \leq b^k$ for all P'_1 from n to the sink, then we do not need to check dominance with respect to resource k.

Strengthened Dominance

Given feasible paths P_1 and P_2 from source to node n, P_1 dominates P_2 whenever:

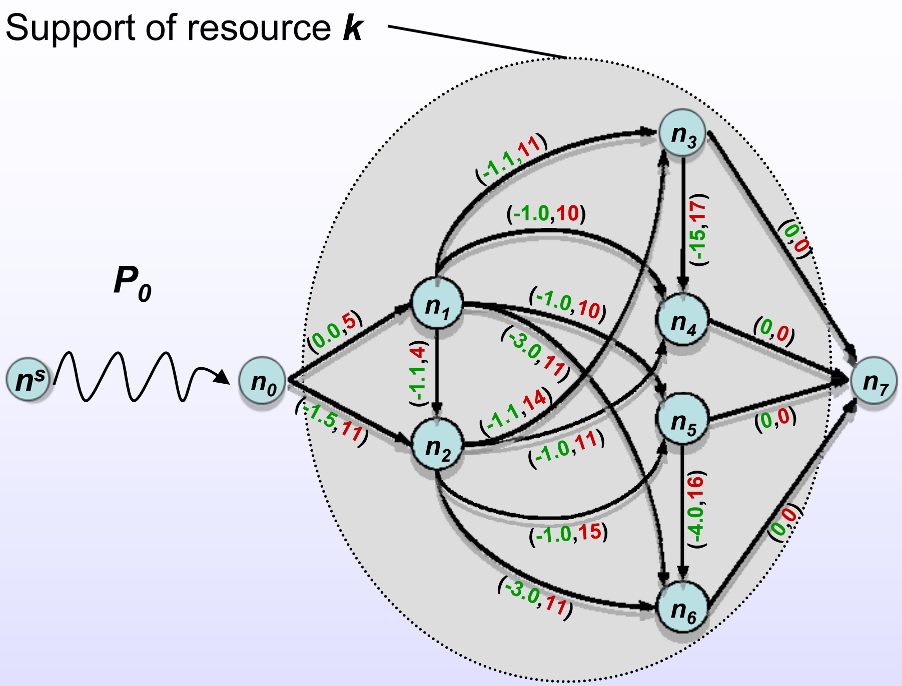
1. $c(P_1) \leq c(P_2)$, and
2. $g^k(P_1) \leq g^k(P_2)$ for all $k = 1,\ldots,K$ s.t. $n \in N^k$.

An Arc-Based Relaxation



path P	$c(P)$	$g^k(P)$
(P_0,n_1)	0.0	5
(P_0,n_2)	-1.5	11
(P_0,n_1,n_2)	-1.1	9
(P_0,n_1,n_3)	-1.1	16
(P_0,n_2,n_3)	-2.6	25
(P_0,n_1,n_2,n_3)	-2.2	23
(P_0,n_1,n_4)	-1.0	15
(P_0,n_2,n_4)	-2.5	22
(P_0,n_1,n_2,n_4)	-2.1	20
(P_0,n_1,n_3,n_4)	-2.6	33
(P_0,n_2,n_3,n_4)	-4.1	42 ✗
(P_0,n_1,n_2,n_3,n_4)	-3.7	40
(P_0,n_1,n_5)	-1.0	15
(P_0,n_2,n_5)	-2.5	26
(P_0,n_1,n_2,n_5)	-2.1	24
(P_0,n_1,n_6)	-3.0	16
(P_0,n_2,n_6)	-4.5	22
(P_0,n_1,n_2,n_6)	-4.1	20
(P_0,n_1,n_5,n_6)	-5.0	31
(P_0,n_2,n_5,n_6)	-6.5	42 ✗
(P_0,n_1,n_2,n_5,n_6)	-6.1	40
No. of feasible states (b^k = 40)		19
No. of non-dominated paths		19

An Arc-Based Relaxation



path P	$c(P)$	$g^k(P)$	$S_5(g^k(P))$
(P_0,n_1)	0.0	5	5
(P_0,n_2)	-1.5	11	10
(P_0,n_1,n_2)	-1.1	9	5
(P_0,n_1,n_3)	-1.1	16	15 ✗
(P_0,n_2,n_3)	-2.6	25	20
(P_0,n_1,n_2,n_3)	-2.2	23	15
(P_0,n_1,n_4)	-1.0	15	15 ✗
(P_0,n_2,n_4)	-2.5	22	20
(P_0,n_1,n_2,n_4)	-2.1	20	15
(P_0,n_1,n_3,n_4)	-2.6	33	30 ✗
(P_0,n_2,n_3,n_4)	-4.1	42 ✗	35
(P_0,n_1,n_2,n_3,n_4)	-3.7	40	30
(P_0,n_1,n_5)	-1.0	15	15
(P_0,n_2,n_5)	-2.5	26	25
(P_0,n_1,n_2,n_5)	-2.1	24	20
(P_0,n_1,n_6)	-3.0	16	15 ✗
(P_0,n_2,n_6)	-4.5	22	20
(P_0,n_1,n_2,n_6)	-4.1	20	15
(P_0,n_1,n_5,n_6)	-5.0	31	30
(P_0,n_2,n_5,n_6)	-6.5	42 ✗	40
(P_0,n_1,n_2,n_5,n_6)	-6.1	40	35
No. of feasible states (b^k = 40)		19	17
No. of non-dominated paths		19	17

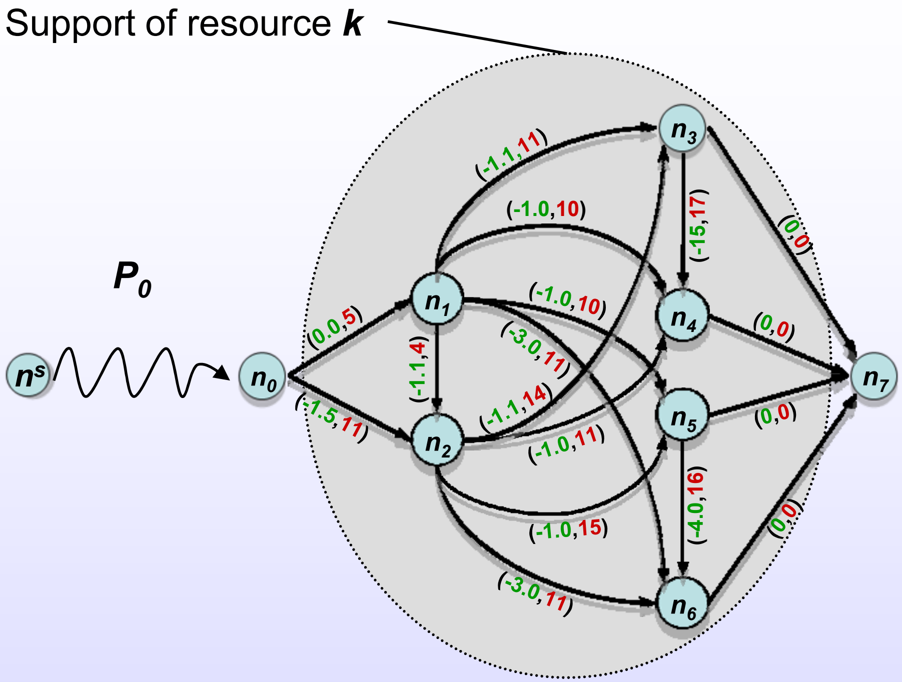
Uniform scaling of resource consumption:

γ - positive integer

$$S_\gamma(g_e^k) = \left\lfloor \frac{g_e^k}{\gamma} \right\rfloor \times \gamma$$

$$S_\gamma(g^k(P)) = \sum_{e \in A(P)} S_\gamma(g_e^k)$$

An Arc-Based Relaxation



path P	$c(P)$	$g^k(P)$	$S_2(g^k(P))$
(P_0,n_1)	0.0	5	4
(P_0,n_2)	-1.5	11	10
(P_0,n_1,n_2)	-1.1	9	8
(P_0,n_1,n_3)	-1.1	16	14
(P_0,n_2,n_3)	-2.6	25	24
(P_0,n_1,n_2,n_3)	-2.2	23	22
(P_0,n_1,n_4)	-1.0	15	14
(P_0,n_2,n_4)	-2.5	22	20
(P_0,n_1,n_2,n_4)	-2.1	20	18
(P_0,n_1,n_3,n_4)	-2.6	33	30
(P_0,n_2,n_3,n_4)	-4.1	42 ✗	40
(P_0,n_1,n_2,n_3,n_4)	-3.7	40	38
(P_0,n_1,n_5)	-1.0	15	14
(P_0,n_2,n_5)	-2.5	26	24
(P_0,n_1,n_2,n_5)	-2.1	24	22
(P_0,n_1,n_6)	-3.0	16	14
(P_0,n_2,n_6)	-4.5	22	20
(P_0,n_1,n_2,n_6)	-4.1	20	18
(P_0,n_1,n_5,n_6)	-5.0	31	30
(P_0,n_2,n_5,n_6)	-6.5	42 ✗	40
(P_0,n_1,n_2,n_5,n_6)	-6.1	40	38
No. of feasible states (b^k = 40)		19	21
No. of non-dominated paths		19	21

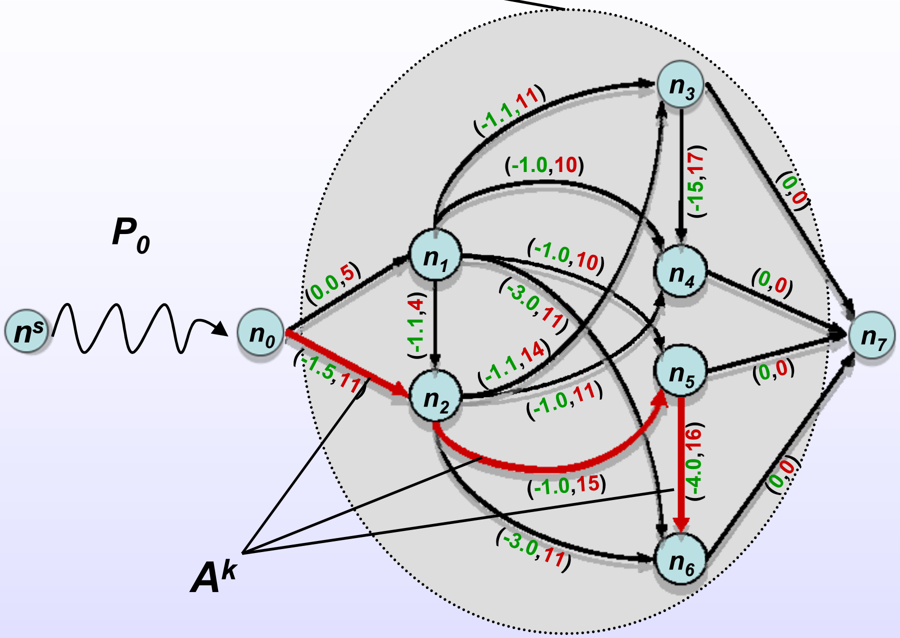
Uniform scaling of resource consumption:

γ - positive integer

$$S_\gamma(g_e^k) = \left\lfloor \frac{g_e^k}{\gamma} \right\rfloor \times \gamma$$

$$S_\gamma(g^k(P)) = \sum_{e \in A(P)} S_\gamma(g_e^k)$$

Support of resource ***k***



An arc based relaxation of resource consumption:

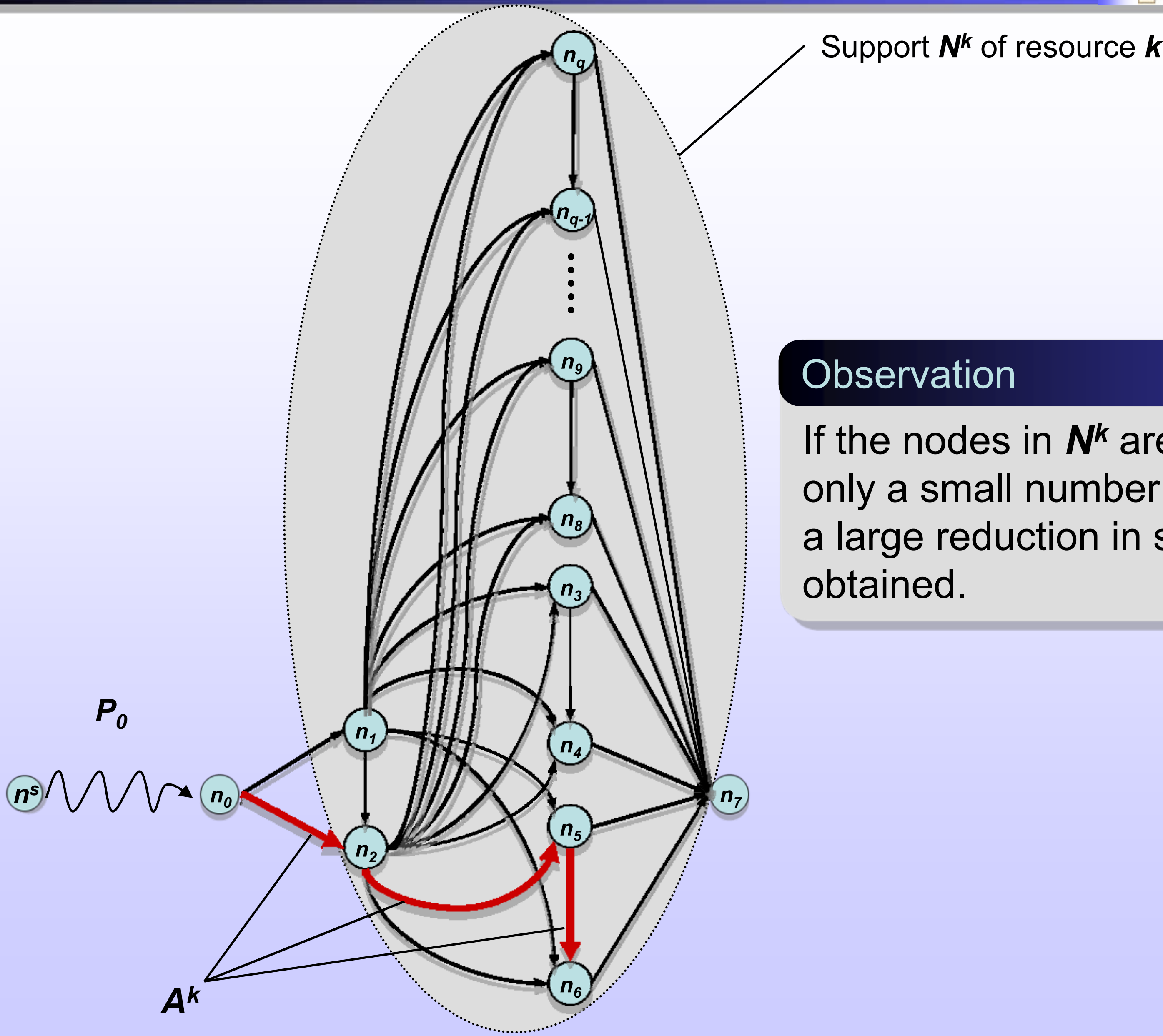
A^k - subset of arcs

$$F_{A^k}(g_e^k) = \begin{cases} g_e^k \text{ if } e \in A^k \\ 0 \text{ otherwise} \end{cases}$$

$$F_{A^k}(g^k(P)) = \sum_{e \in A(P)} F_{A^k}(g_e^k)$$

path P	$c(P)$	$g^k(P)$	$S_2(g^k(P))$	$F_{A^k}(g^k(P))$
(P_0,n_1)	0.0	5	4	0
(P_0,n_2)	-1.5	11	10	10
(P_0,n_1,n_2)	-1.1	9	8	0
(P_0,n_1,n_3)	-1.1	16	14	0 ✗
(P_0,n_2,n_3)	-2.6	25	24	11
(P_0,n_1,n_2,n_3)	-2.2	23	22	0
(P_0,n_1,n_4)	-1.0	15	14	0 ✗
(P_0,n_2,n_4)	-2.5	22	20	11 ✗
(P_0,n_1,n_2,n_4)	-2.1	20	18	0 ✗
(P_0,n_1,n_3,n_4)	-2.6	33	30	0 ✗
(P_0,n_2,n_3,n_4)	-4.1	42 ✗	40	11
(P_0,n_1,n_2,n_3,n_4)	-3.7	40	38	0
(P_0,n_1,n_5)	-1.0	15	14	0
(P_0,n_2,n_5)	-2.5	26	24	26
(P_0,n_1,n_2,n_5)	-2.1	24	22	15
(P_0,n_1,n_6)	-3.0	16	14	0 ✗
(P_0,n_2,n_6)	-4.5	22	20	11
(P_0,n_1,n_2,n_6)	-4.1	20	18	0
(P_0,n_1,n_5,n_6)	-5.0	31	30	16
(P_0,n_2,n_5,n_6)	-6.5	42 ✗	40	42 ✗
(P_0,n_1,n_2,n_5,n_6)	-6.1	40	38	31
No. of feasible states (b^k = 40)		19	21	14
No. of non-dominated paths		19	21	14

Georgia Institute of Technology
The H. Milton Stewart School of Industrial and Systems Engineering



Observation

If the nodes in N^k are reachable from only a small number of arcs in A^k, then a large reduction in state-space can be obtained.

A Relaxation based DP

An iterative relaxation based search procedure

Input : Network $\mathcal{N} = (N, A)$, source n^s, and sink n^t;
c_e for all $e \in A$ and g_e^k for all $e \in A$ and $k \in \{1,\ldots,K\}$;

Initialize : $UB \leftarrow \infty$ and $LB \leftarrow \infty$;
$A^k \leftarrow \{\phi\}$ for all $k \in \{1,\ldots,K\}$;

```
while UB − LB > ε
    (P*,UB) = DP(𝒩, A^1,…A^K)
    LB ← c(P*)
    if P* is infeasible then
        add arcs to A^k for one or more k so that P* is no
        longer feasible in relaxation
    end
end
```

Output : P^*

Ignore all resource consumption.

Run DP to find min-cost path from source to sink when only keeping track of resource *k* on arcs in A^k for each $k \in \{1,\ldots,K\}$.

Update *LB*.

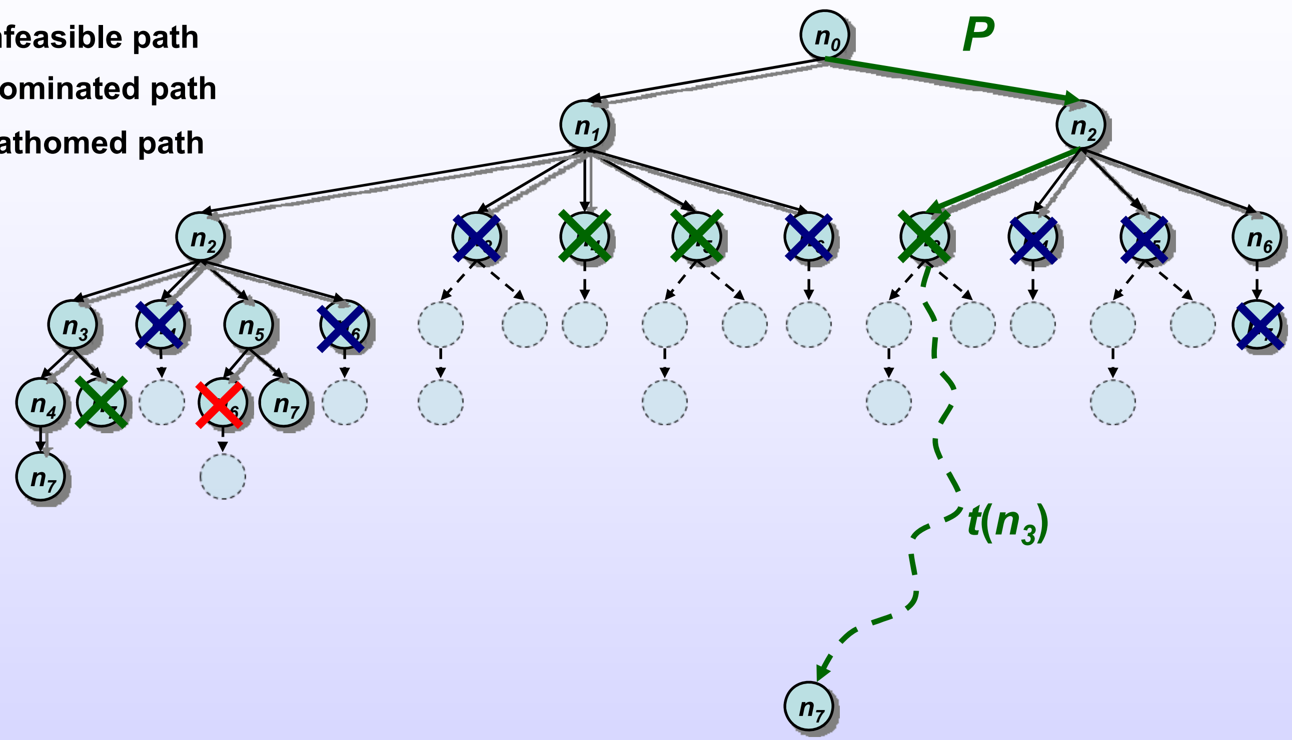
Refine relaxation by adding arcs to A^k for one or more *k*.

Bounding the search

Georgia Institute of Technology
The H. Milton Stewart School of Industrial and Systems Engineering

The Search Tree:

- Infeasible path
- Dominated path
- Fathomed path



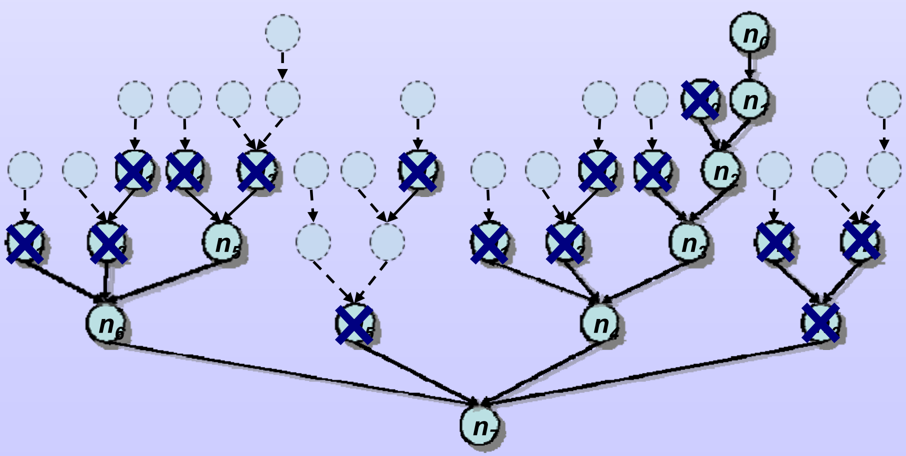
Let $t(n)$ be a lower-bound on the cost of a min-cost path from n to the sink.

$\Rightarrow$ If $c(P) + t(n_3) > UB$, then P can be discarded (i.e. search fathomed at P)

Q: How to compute $t(n)$ for each n?

A: Bounds obtained for FREE as a natural byproduct of relaxation scheme and alternating search directions.

The Backward Search Tree:

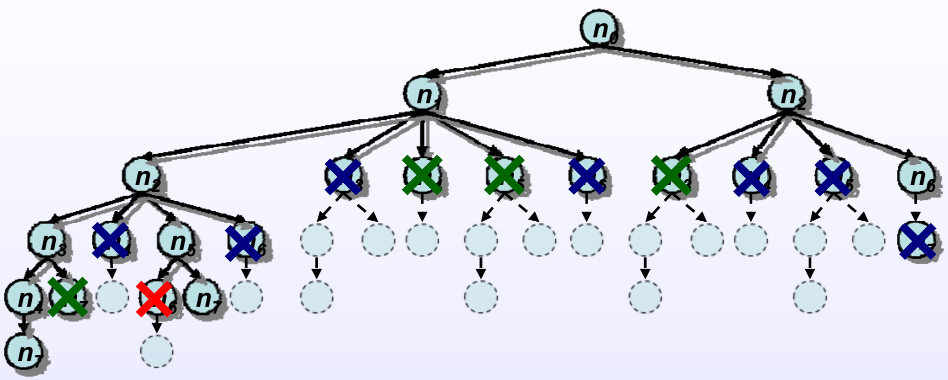


$$t(n) = \min\left\{ c(P) : \begin{array}{l} P \text{ is a non-dominated path} \\ \text{from } n \text{ to sink.} \end{array} \right\}$$

***t*(*n*) is a valid bound for pruning search from source to sink.**

Computing Bounds for Fathoming

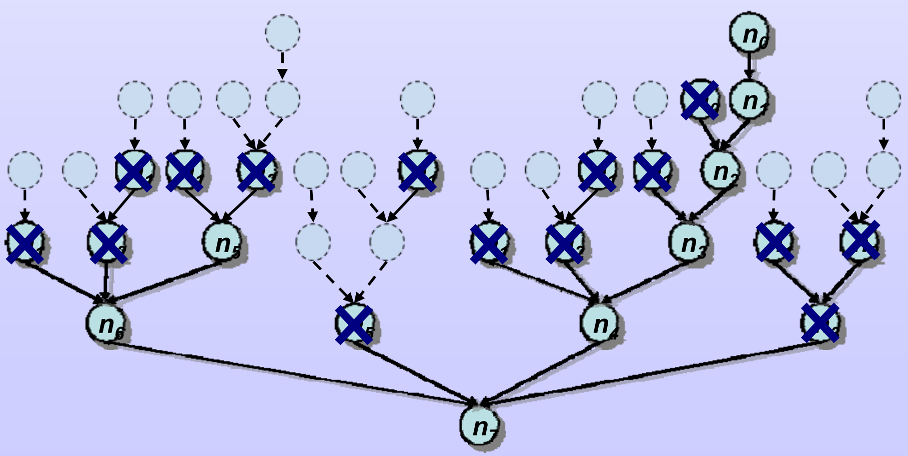
The Forward Search Tree:



$$s(n) = \min\left\{ c(P) : \begin{array}{l} P \text{ is a non-dominated path} \\ \text{from source to } n. \end{array} \right\}$$

***s*(*n*) is a valid bound for pruning search from sink to source.**

The Backward Search Tree:



$$t(n) = \min\left\{ c(P) : \begin{array}{l} P \text{ is a non-dominated path} \\ \text{from } n \text{ to sink.} \end{array} \right\}$$

***t*(*n*) is a valid bound for pruning search from source to sink.**

A Relaxation + Bounding based DP

An iterative relaxation + bounding search procedure

Input : Network $\mathcal{N} = (N, A)$, source n^s, and sink n^t;
c_e for all $e \in A$ and g_e^k for all $e \in A$ and $k \in \{1,\ldots,K\}$;

Initialize : $UB \leftarrow \infty$ and $LB \leftarrow \infty$;
$A^k \leftarrow \{\phi\}$ for all $k \in \{1,\ldots,K\}$;
$s(n) \leftarrow -\infty$ and $t(n) \leftarrow -\infty$ for all $n \in N \setminus \{n^s, n^t\}$;
$s(n^s) \leftarrow 0$ and $t(n^t) \leftarrow 0$;

while $UB - LB > \varepsilon$
  if *pass* is even **then**
    $(P^*, UB, s(n)\ \forall n \in N) =$ **FwdDP**$(\mathcal{N}, A^1, \ldots A^K, t(n)\ \forall n \in N)$
  else
    $(P^*, UB, t(n)\ \forall n \in N) =$ **BwdDP**$(\mathcal{N}, A^1, \ldots A^K, s(n)\ \forall n \in N)$
  end
  $LB \leftarrow c(P^*)$
  if P^* is infeasible **then**
    add arcs to A^k for one or more k so that P^* is no longer feasible in relaxation
  end
end

Output : P^*

Ignore all resource consumption.

Initialize bounds for fathoming.

Alternate between Forward and Backward searches only keeping track of resource k on arcs in A^k for each $k \in \{1,\ldots,K\}$.

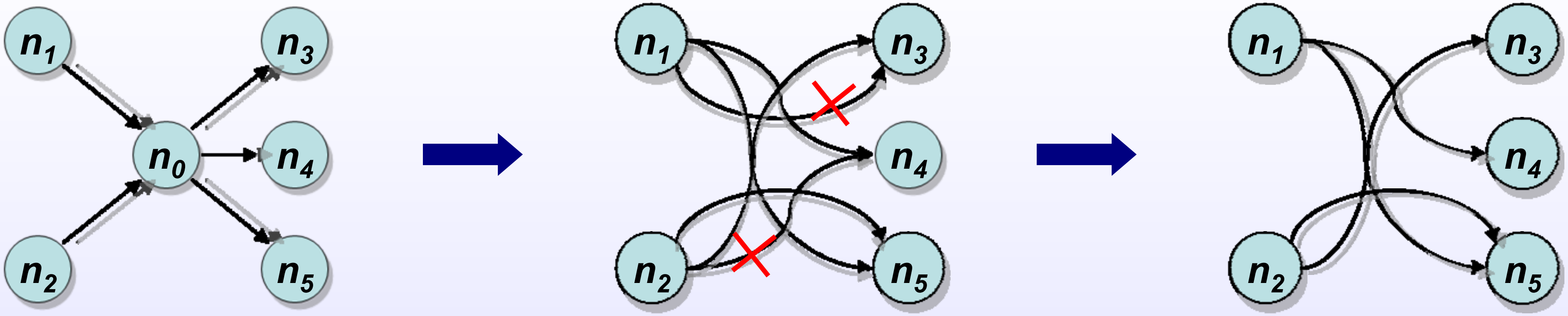
Update bounds for fathoming.

Update LB.

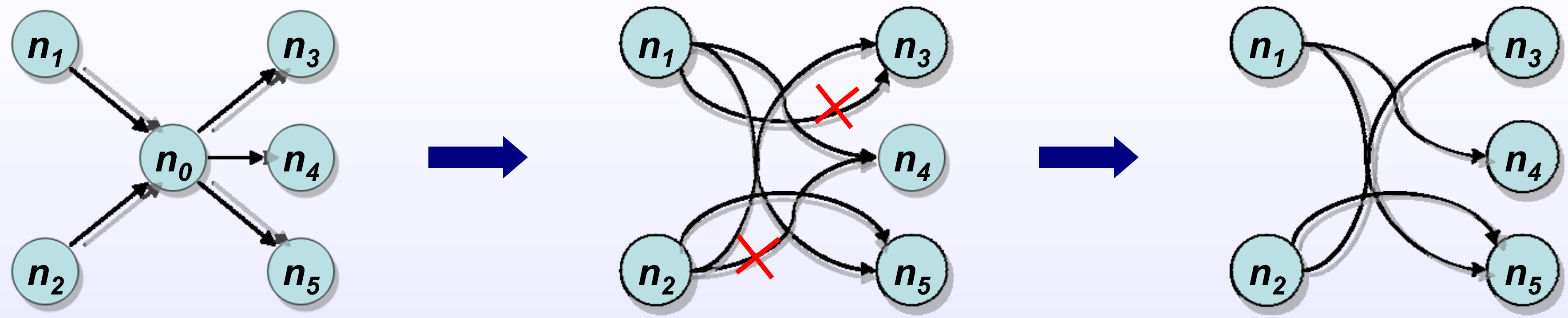
Refine relaxation by adding arcs to A^k for one or more k.

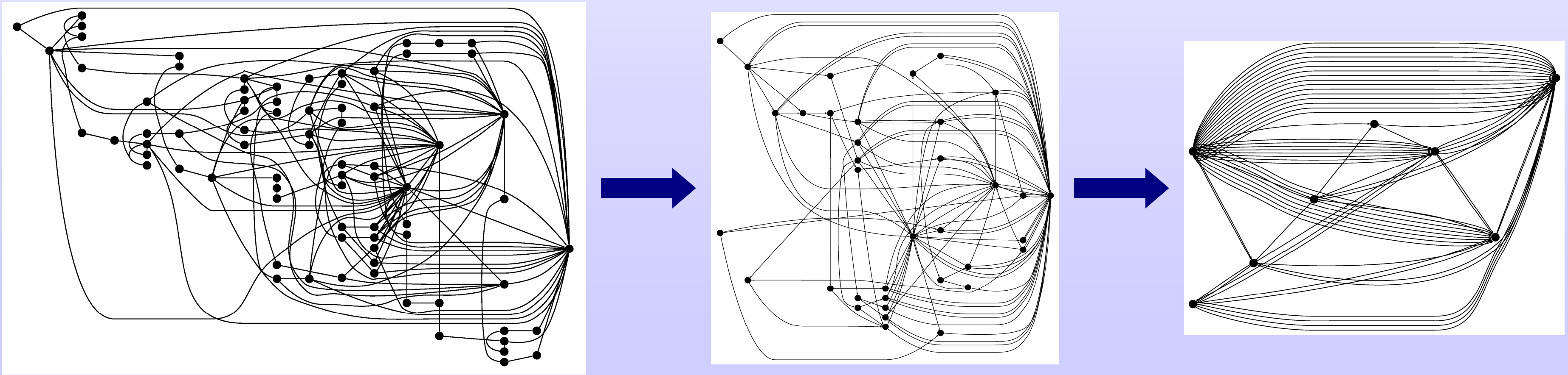
Preprocessing by Aggregation

Georgia Institute of Technology
The H. Milton Stewart School of Industrial and Systems Engineering



Preprocessing by Aggregation





Impact of Aggregation on Network

no. of nodes

min, max, and average size of supports

No. Jets	No. Ports	No. Reqs	Agg.	$\|N\|$ ($\times 10^6$)	$\|A\|$ ($\times 10^6$)	K ($\times 10^6$)	min(N^k)	max(N^k) ($\times 10^6$)	avg(N^k)
10	15	60	No	0.22	0.37	0.24	1	0.11	10
			Yes	0.01	0.11	0.14	1	0.004	2
25	20	173	No	0.52	0.97	0.50	2	0.26	24
			Yes	0.03	0.38	0.35	1	0.014	3
50	30	356	No	1.24	2.50	1.05	2	0.62	56
			Yes	0.09	1.14	0.84	1	0.043	8
75	30	559	No	1.94	4.11	1.33	2	0.97	105
			Yes	0.15	2.04	1.13	1	0.073	15
100	35	751	No	2.79	6.22	1.69	4	1.39	160
			Yes	0.24	3.33	1.51	1	0.12	24
125	41	956	No	3.77	8.67	2.00	5	1.89	240
			Yes	0.34	4.89	1.88	2	0.17	36
150	41	1172	No	4.60	10.80	2.11	5	2.30	328
			Yes	0.41	6.20	2.01	1	0.20	47
175	41	1364	No	5.56	13.33	2.23	6	2.78	429
			Yes	0.50	7.75	2.13	2	0.25	62
185	41	1448	No	5.90	14.28	2.25	5	2.95	474
			Yes	0.54	8.39	2.17	2	0.27	70
200	41	1457	No	6.40	15.64	2.28	6	3.2	534
			Yes	0.58	9.23	2.20	2	0.29	78

Computational Tests

The H. Milton Stewart School of Industrial and Systems Engineering

Refinement schemes

S^0 – Standard DP with proposed dominance scheme.

S^1 – Start with relaxing all resource consumption and refine relaxation over a large number of resources and arcs.

S^2 – Start as in **S^1** and then switch to a more conservative strategy by only tracking resources and arcs that are part of infeasible paths that are used to compute bounds for pruning.

Computational Tests

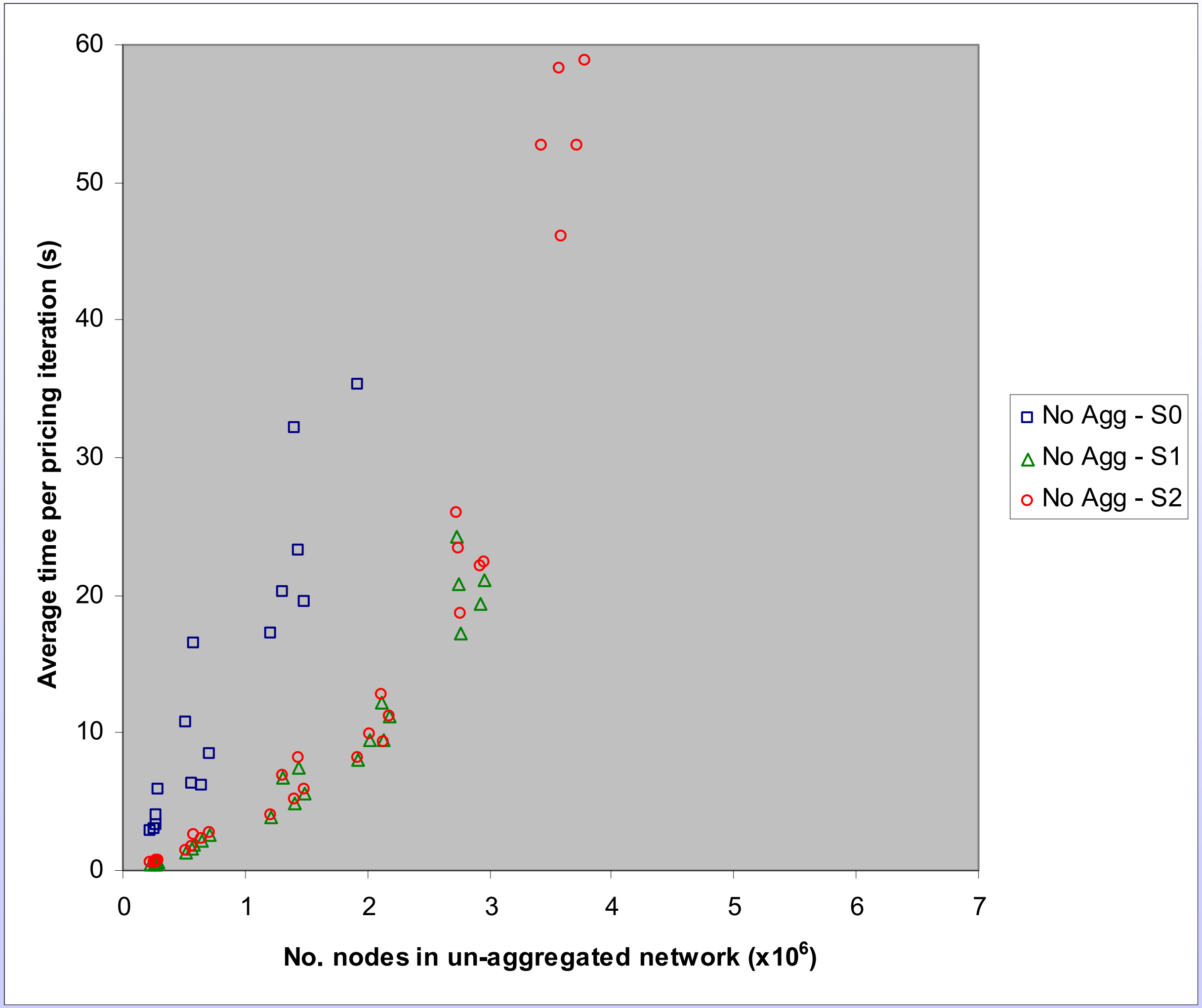
Refinement schemes

S^0 – Standard DP with proposed dominance scheme.

S^1 – Start with relaxing all resource consumption and refine relaxation over a large number of resources and arcs.

S^2 – Start as in **S^1** and then switch to a more conservative strategy by only tracking resources and arcs that are part of infeasible paths that are used to compute bounds for pruning.

Impact on speed



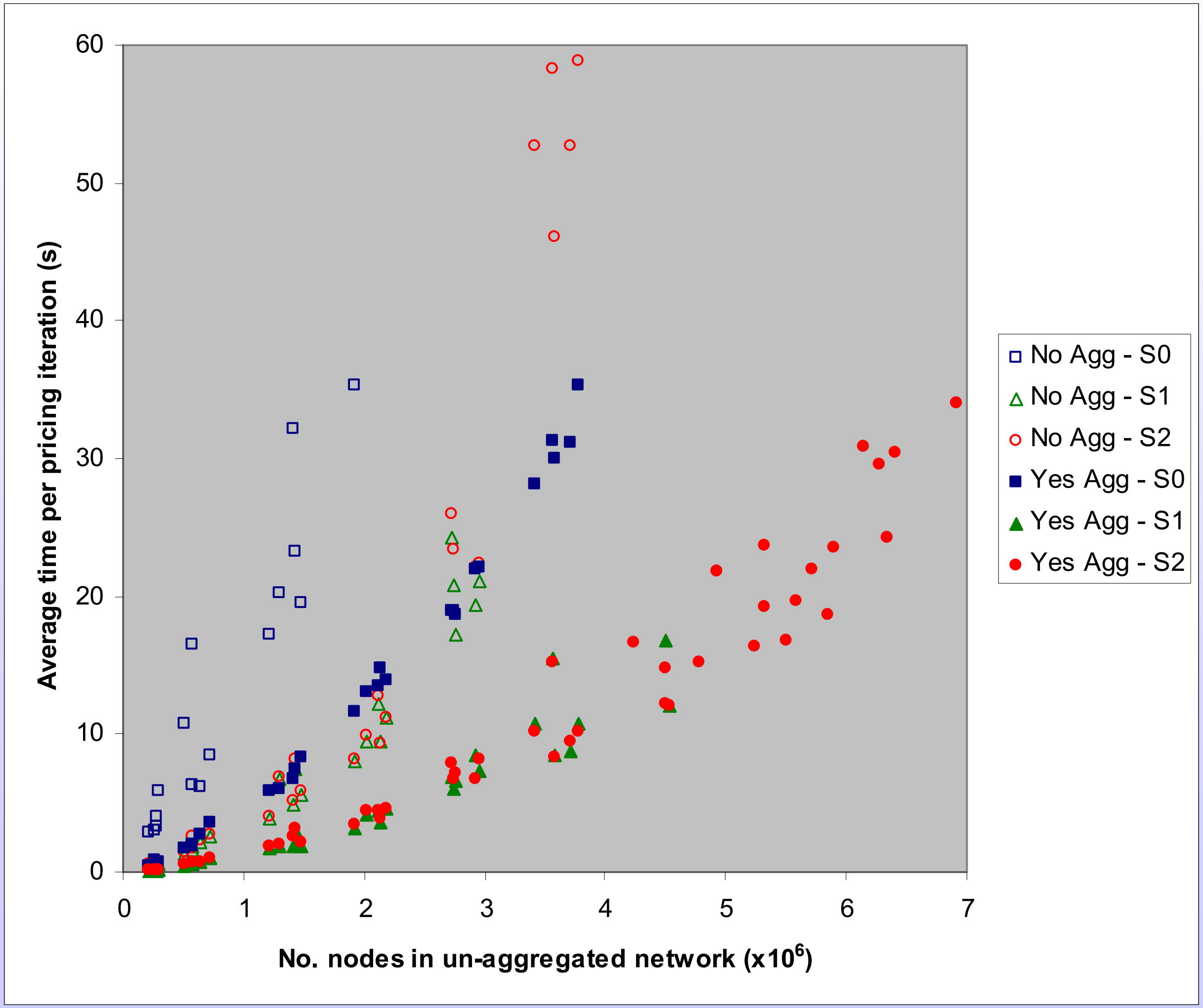
Refinement schemes

S⁰ – Standard DP with proposed dominance scheme.

S¹ – Start with relaxing all resource consumption and refine relaxation over a large number of resources and arcs.

S² – Start as in **S¹** and then switch to a more conservative strategy by only tracking resources and arcs that are part of infeasible paths that are used to compute bounds for pruning.

Impact on speed



Computational Tests

Georgia Institute of Technology
The H. Milton Stewart School of Industrial and Systems Engineering

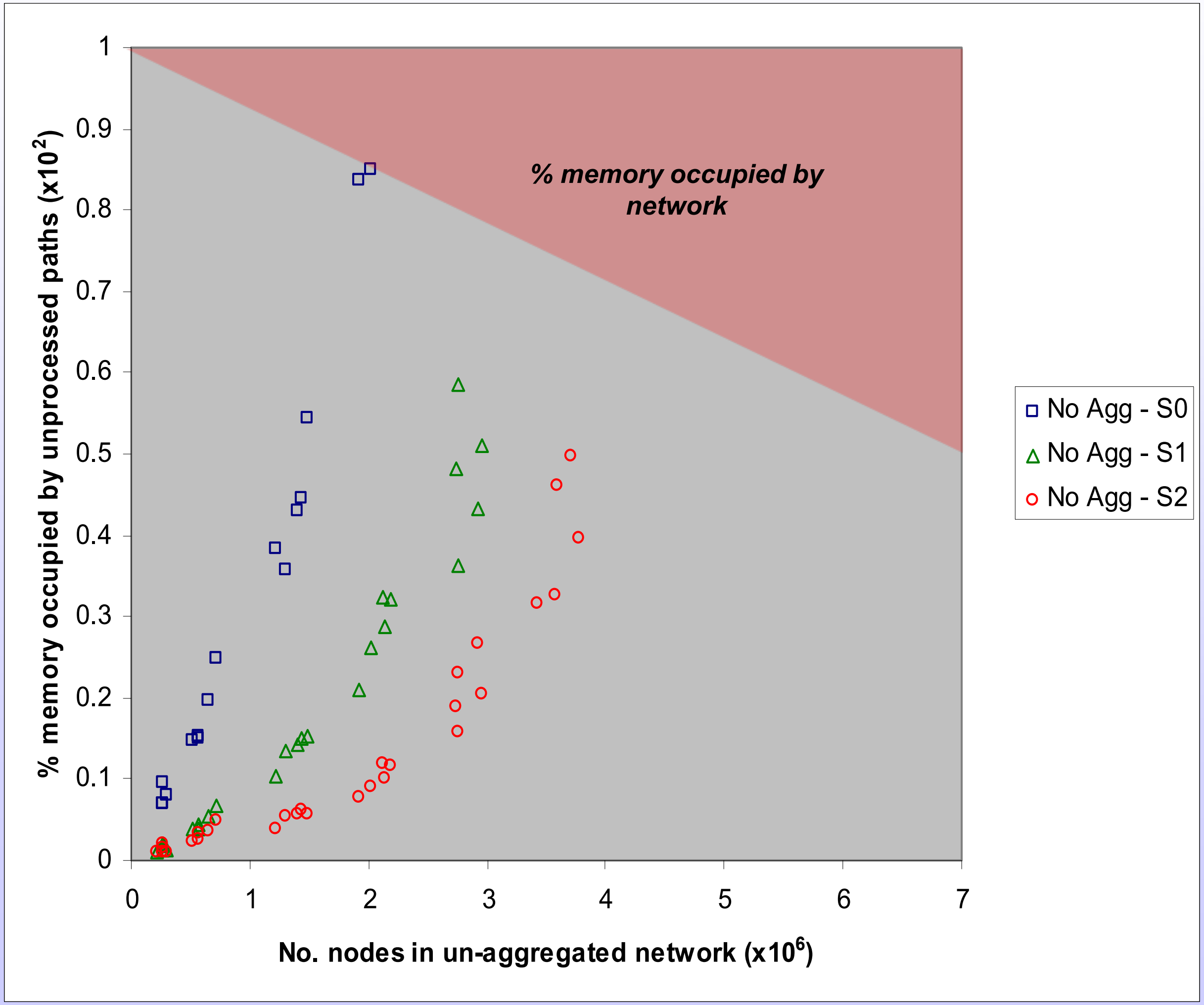
Refinement schemes

S⁰ – Standard DP with proposed dominance scheme.

S¹ – Start with relaxing all resource consumption and refine relaxation over a large number of resources and arcs.

S² – Start as in **S¹** and then switch to a more conservative strategy by only tracking resources and arcs that are part of infeasible paths that are used to compute bounds for pruning.

Impact on memory requirement



Computational Tests

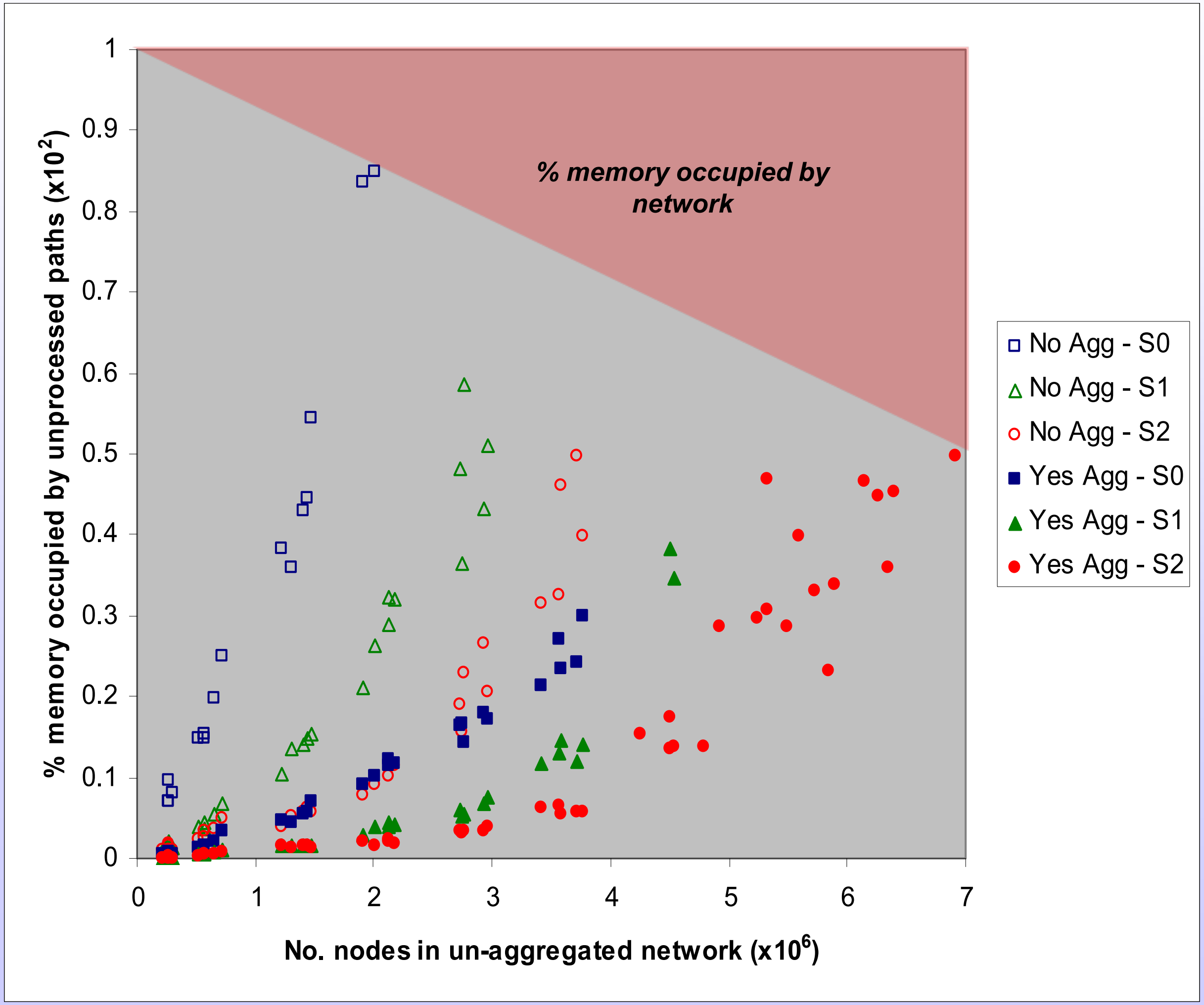
Refinement schemes

S^0 – Standard DP with proposed dominance scheme.

S^1 – Start with relaxing all resource consumption and refine relaxation over a large number of resources and arcs.

S^2 – Start as in **S^1** and then switch to a more conservative strategy by only tracking resources and arcs that are part of infeasible paths that are used to compute bounds for pruning.

Impact on memory requirement



Refinement schemes

S^0 – Standard DP with proposed dominance scheme.

S^1 – Start with relaxing all resource consumption and refine relaxation over a large number of resources and arcs.

S^2 – Start as in **S^1** and then switch to a more conservative strategy by only tracking resources and arcs that are part of infeasible paths that are used to compute bounds for pruning.

No. Jets	No. Ports	No. Requests	No Agg Scheme: S^0	No Agg Scheme: S^1	No Agg Scheme: S^2	Yes Agg Scheme: S^0	Yes Agg Scheme: S^1	Yes Agg Scheme: S^2
10	15	[60,80]	5	5	5	5	5	5
			3.06	0.52	0.63	0.66	0.08	0.13
			0.00	2.25	3.95	0.00	2.12	3.58
			0.89	0.18	0.16	0.07	0.02	0.01
25	20	[173,199]	5	5	5	5	5	5
			7.75	1.97	2.21	2.45	0.67	0.77
			0.00	2.55	4.05	0.00	2.33	3.86
			2.15	0.58	0.41	0.23	0.07	0.06
50	30	[350,369]	5	5	5	5	5	5
			19.42	5.71	5.94	6.89	2.11	2.35
			0.00	2.78	5.09	0.00	2.48	4.74
			5.19	1.64	0.65	0.65	0.19	0.17
75	30	[543,585]	2	5	5	5	5	5
			33.83	10.04	10.21	13.84	3.42	3.54
			0.00	2.88	5.23	0.00	2.75	4.97
			10.11	3.38	1.21	1.32	0.46	0.23
100	35	[746,779]	0	5	5	5	5	5
			-	20.78	22.68	20.15	7.29	7.36
			-	2.87	5.52	0.00	2.51	5.23
			-	5.70	2.52	1.98	0.74	0.42
125	41	[934,966]	0	0	5	5	5	5
			-	-	53.87	31.19	10.86	10.74
			-	-	6.12	0.00	2.57	5.52
			-	-	4.79	3.03	1.57	0.72
150	41	[1135,1182]	0	0	0	0	2	5
			-	-	-	-	14.26	14.13
			-	-	-	-	2.68	5.64
			-	-	-	-	4.37	1.78
175	41	[1312,1382]	0	0	0	0	0	5
			-	-	-	-	-	18.56
			-	-	-	-	-	5.97
			-	-	-	-	-	3.38
185	41	[1385,1487]	0	0	0	0	0	5
			-	-	-	-	-	22.75
			-	-	-	-	-	6.31
			-	-	-	-	-	4.55
200	41	[1516,1613]	0	0	0	0	0	4
			-	-	-	-	-	31.26
			-	-	-	-	-	6.97
			-	-	-	-	-	5.60

No. of solved instances

Average time (s) per pricing iteration.

Average no. of refinements per pricing iteration.

Max no. of stored paths ($x10^6$)

Questions?